# Precipitation Nowcasting in the Canary Islands with Deep Learning Tools at EWCloud

David Quintero Plaza, AEMET, Spain. EWCloud Meeting, November 10th, 2021
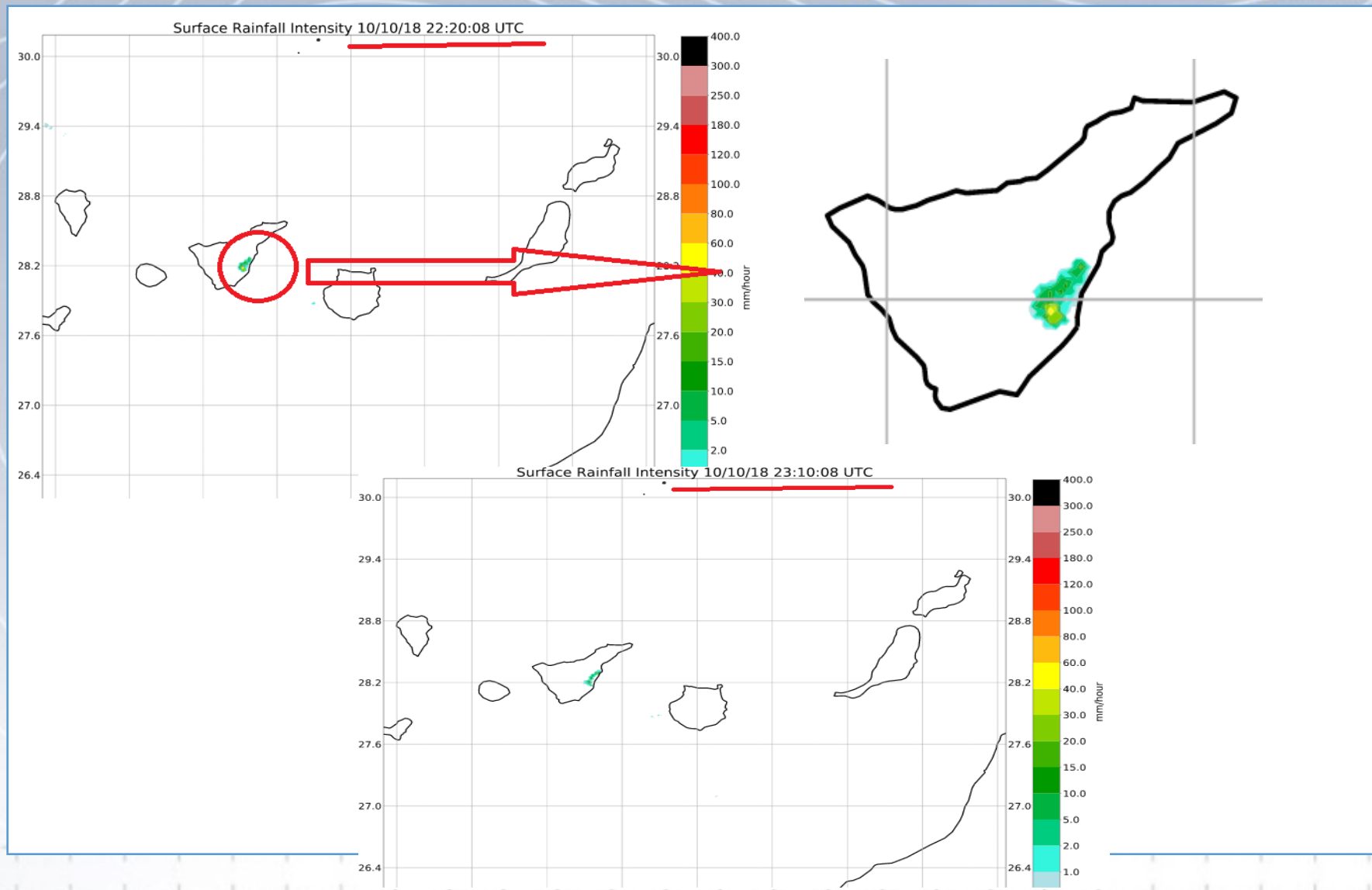
Index.

1. The Canary Islands: a challenge for modelization.

2. Deep learning for precipitation nowcasting.

      2.1. EWCloud setup.

      2.2. First trainings.

3. A glimpse to the future.

- The islands have a complex, mountainous and fastly changing terrain, with the so called local "micro-climates".
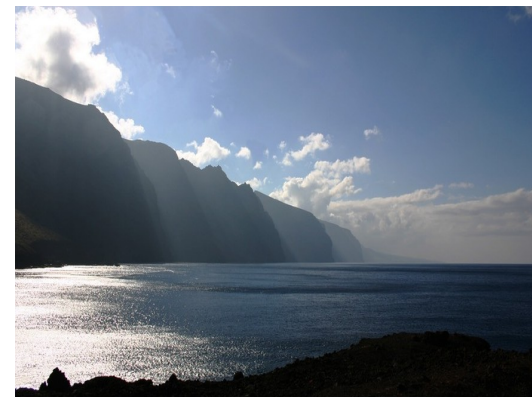
- Even High Resolution Models (Harmonie & gSREPS at 2.5 Km) still show limitations for the islands (not to mention models with less resolution).

- Affected by middle latitudes and tropical phenomena.

Surface Rainfall Intensity 10/10/18 22:20:08 UTC

Surface Rainfall Intensity 10/10/18 23:10:08 UTC

- Regarding the nowcasting: 3 possibilities,

- Fast Integration Models.

- Optical Flow.

- **Neural Networks and other Machine Learning approaches.**

- Some evidence that Machine Learning/Deep Learning is on equal terms (and sometimes beating) more *classical* options...

MetNet: A Neural Weather Model for Precipitation Forecasting. https://arxiv.org/abs/2003.12140

Convcast: An embedded convolutional LSTM based architecture for precipitation nowcasting using satellite data. PLoS ONE 15(3): e0230114. https://doi.org/10.1371/journal.pone.0230114

All convolutional neural networks for radar-based precipitation nowcasting. Procedia Computer Science Volume 150, 2019, Pages 186-192. https://doi.org/10.1016/j.procs.2019.02.036

- Working with the radar.

- The variable chosen was **SRI (Surface Rainfall Intensity)**, an estimated value of the instantaneous precipitation using data from all the vertical, as offered by the IRIS SIGMET-VAISALA radar, **each 10 minutes, in mm/hour**.

- Beware of the limitations: radar located at 1778 meters above sea level & many holes in the historical dataset!

- User environment at the EWCloud…

- *conda* installation of different libraries, especially NumPy, **Tensorflow for GPU (tf-gpu=2.3, cudnn=7.6, cudatoolkit=10.1)** and wradlib (for radar data reading).

- Algorithm chosen: **RainNet, by Georgy Ayzel:** a combination of ResNet and U-Net neural networks.

Ayzel, G., Scheffer, T., and Heistermann, M.: RainNet v1.0: a convolutional neural network for radar-based precipitation nowcasting, Geosci. Model Dev., 13, 2631–2644, https://doi.org/10.5194/gmd-13-2631-2020, 2020.

https://github.com/hydrogo/rainnet

- RainNet takes the last 4 images and predicts the next one (5th). Then, it takes the "new" last 4 images (including the predicted before) and predicts the 6th, and so on...

- Original work: German radar with pcp accumulations each 5 minutes. Our radar: accumulations per hour and each 10 minutes => transform and interpolate. **Important step**.
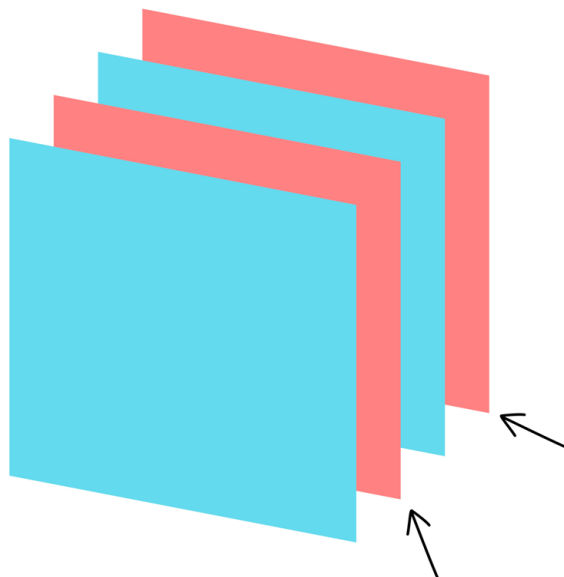
- German dataset: 900x900 matrices converted to 928x928 (U-Net needs rows and cols multiple of 2^{n+1}, n number of max pooling layers, 4 in the original design). Our case: 480x480, already multiple of 32.

- A glimpse into the code... more than 31 millions of parameters to train in our case:

```
inputs = Input(input_shape)

conv1f = Conv2D(64, 3, padding='same', kernel_initializer='he_normal')(inputs)
conv1f = Activation("relu")(conv1f)
conv1s = Conv2D(64, 3, padding='same', kernel_initializer='he_normal')(conv1f)
conv1s = Activation("relu")(conv1s)
pool1 = MaxPooling2D(pool_size=(2, 2))(conv1s)

conv2f = Conv2D(128, 3, padding='same', kernel_initializer='he_normal')(pool1)
conv2f = Activation("relu")(conv2f)
conv2s = Conv2D(128, 3, padding='same', kernel_initializer='he_normal')(conv2f)
conv2s = Activation("relu")(conv2s)
pool2 = MaxPooling2D(pool_size=(2, 2))(conv2s)

conv3f = Conv2D(256, 3, padding='same', kernel_initializer='he_normal')(pool2)
conv3f = Activation("relu")(conv3f)
conv3s = Conv2D(256, 3, padding='same', kernel_initializer='he_normal')(conv3f)
conv3s = Activation("relu")(conv3s)
pool3 = MaxPooling2D(pool_size=(2, 2))(conv3s)
```

- It needs GPUs!

- Our dataset has holes.

- Training has to be done with a temporal set of files without holes (temporal ordering is important).

- Workflow:

1) Download historical data (SRI) from the radar until a hole appears.

2) Do the necessary transformations on the dataset downloaded in 1.

3) Training (early stopping, model checkpoint).

4) Go back to 1 and repeat.

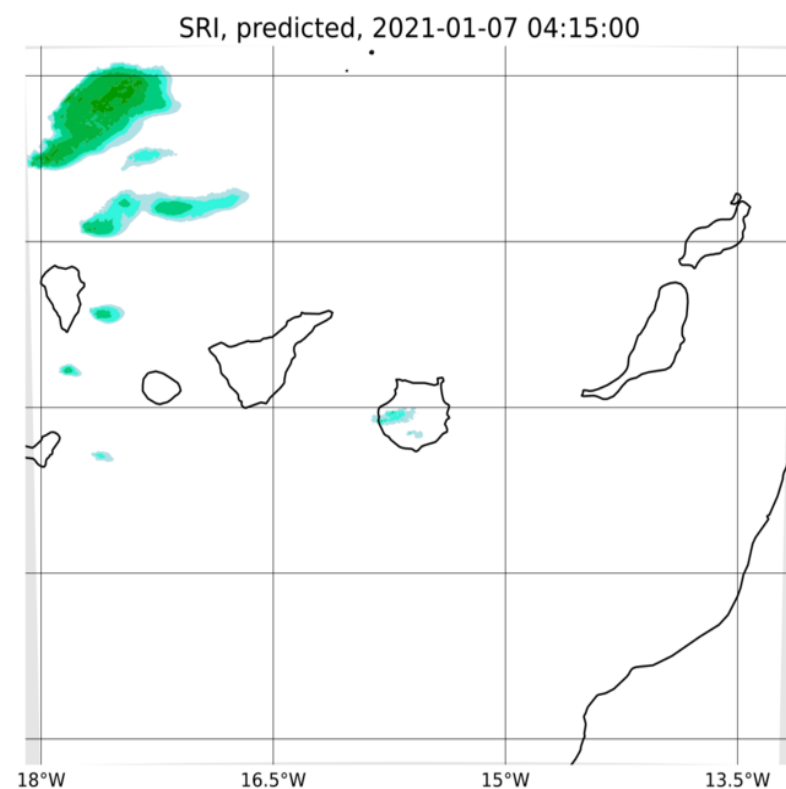**- A first validation...** (with the training set, not valid!! It is just to show).

T+10 minutes:
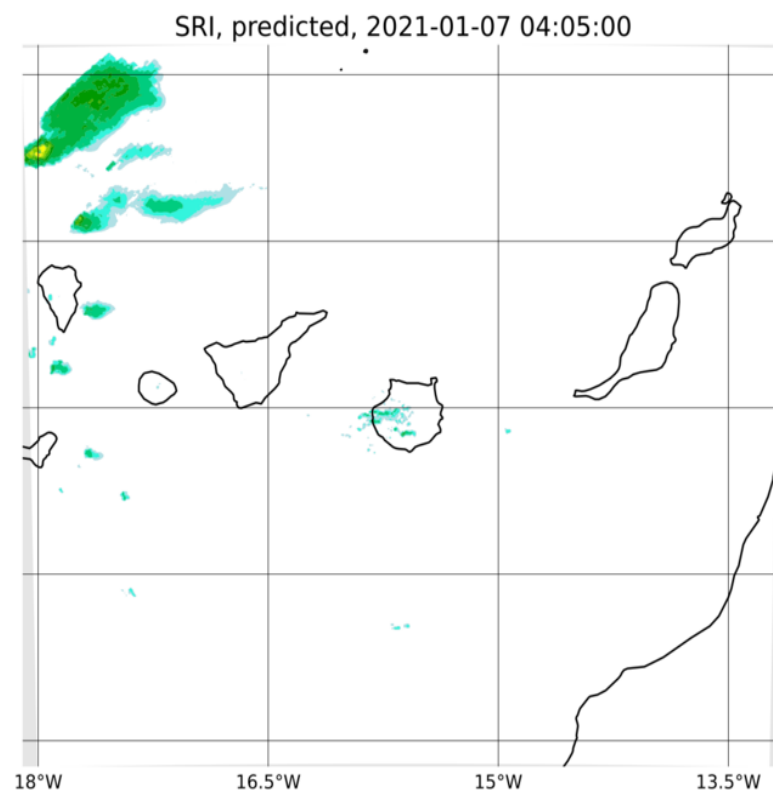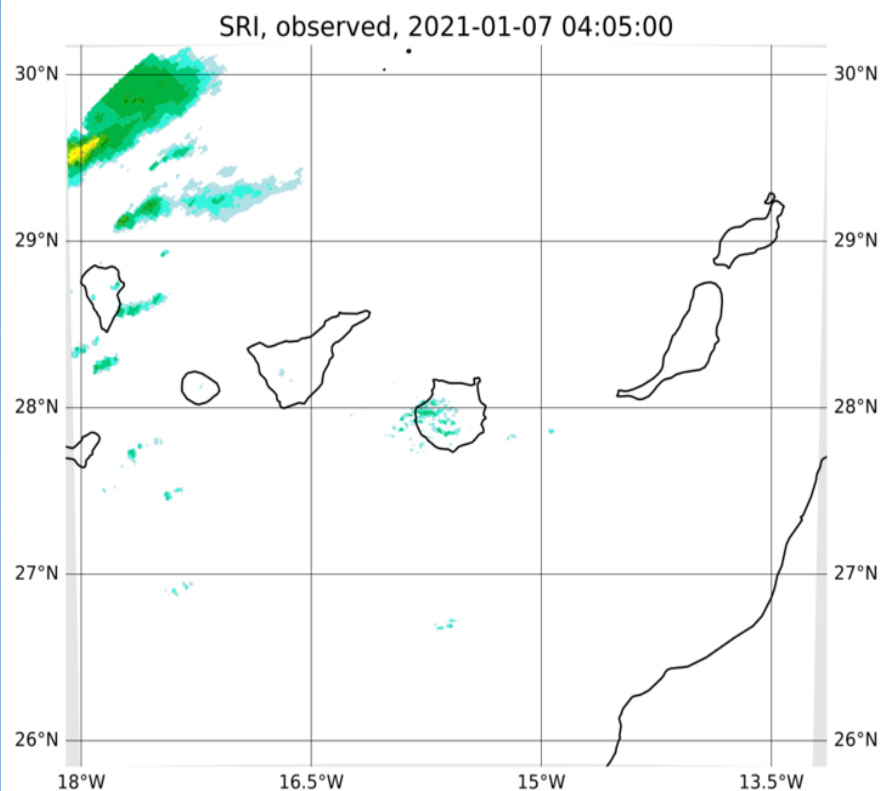
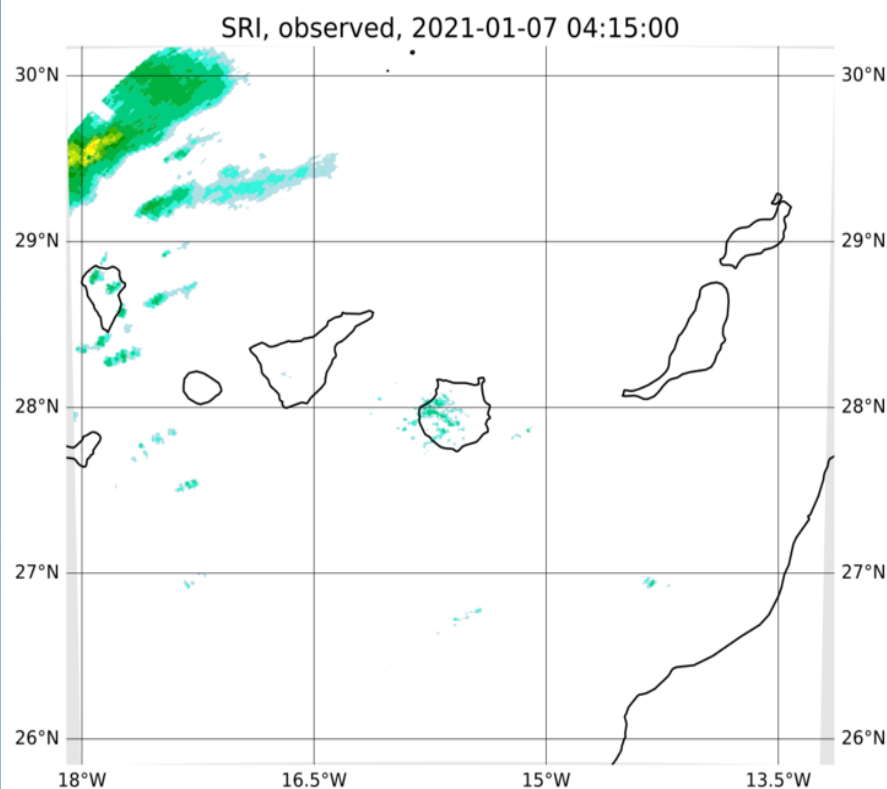**- A first validation...** (with the training set, not valid!! It is just to show).

T+20 minutes:

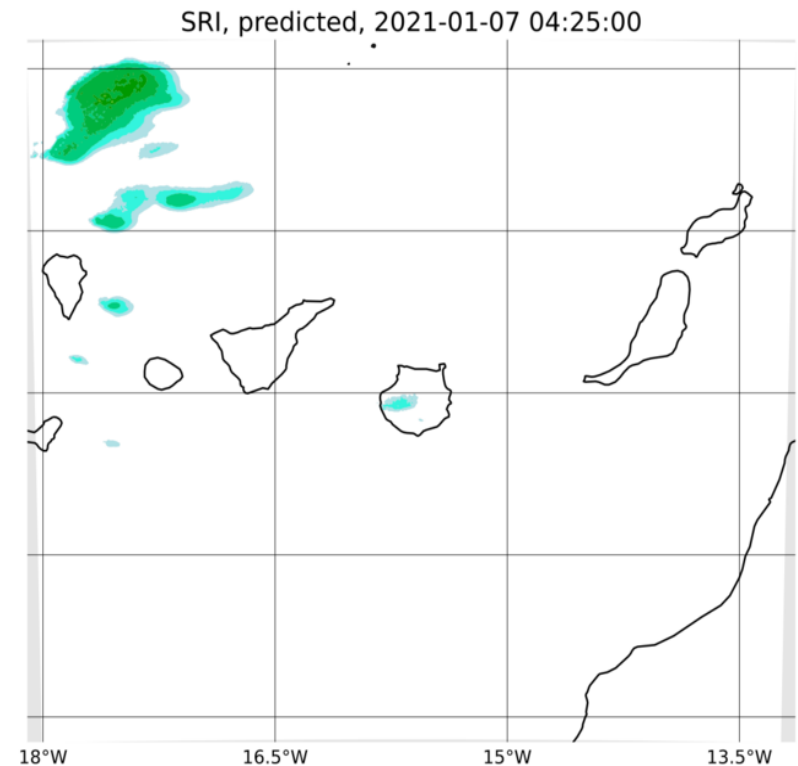**- A first validation...** (with the training set, not valid!! It is just to show).

T+30 minutes:
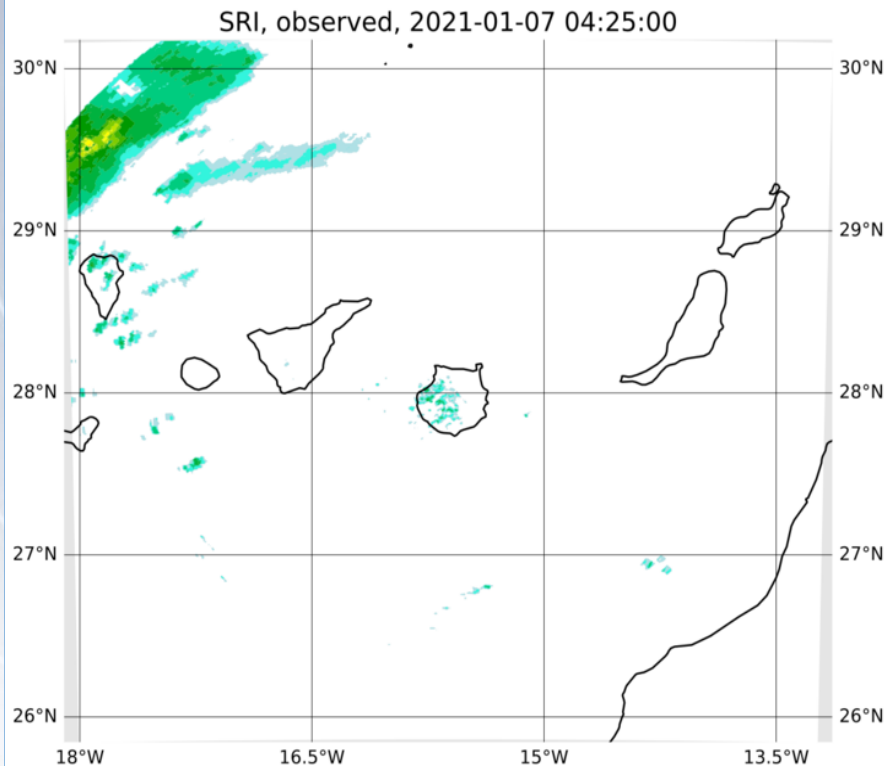
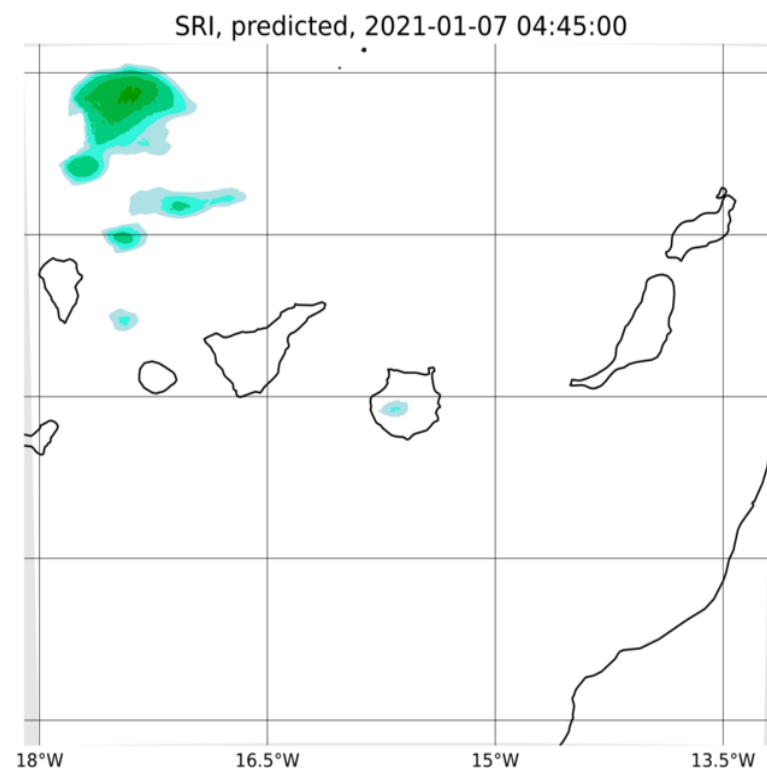SRI, observed, 2021-01-07 04:25:00

SRI, predicted, 2021-01-07 04:25:00

**- A first validation...** (with the training set, not valid!! It is just to show).

T+50 minutes:
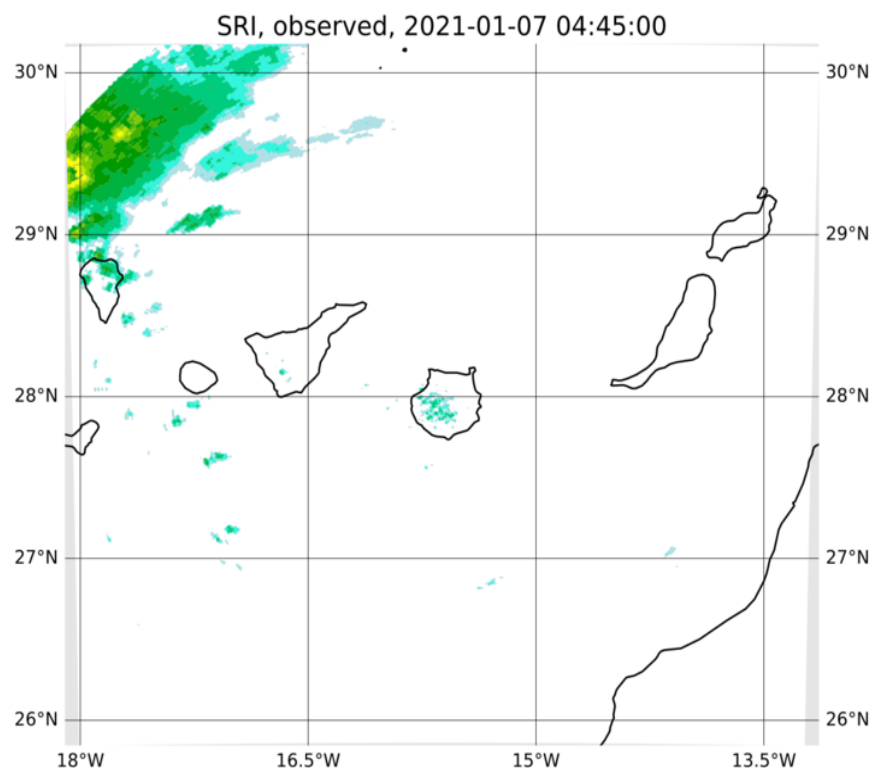
- A glimpse to the future:

- Fix the drastic underestimation of the prediction when time increases. (One NN for each timestep? Other NN architectures? Increase training dataset?...)

- From test to the operative chain.

- Make tests with ConvLSTM neurons as suggested by part of the literature.

- Train in other areas besides the Canary Islands? Wetter regions?

- From radar to satellite?

- More exotic NN architectures (e.g. GANs)?
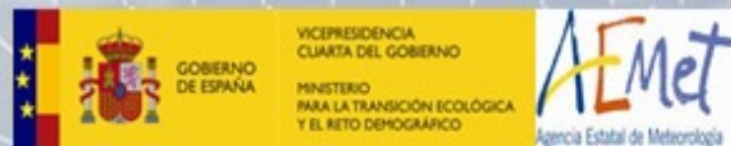
References:

P. 3: Left image: By Iven Gummelt - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=4040284

Central image: By Jens Steckert - Own work, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=765158

Right image: By Daniel Gaínza (Tenerife) - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=4555690

P. 7: Icon EWCloud from https://www.europeanweather.cloud/

Icon RainNet from https://github.com/hydrogo/rainnet

P. 9: From https://github.com/hydrogo/rainnet/blob/master/rainnet.py